SIGEVOlution

newsletter of the ACM Special Interest Group on Genetic and Evolutionary Computation

Winter 2008
Volume 3 Issue 4

in this issue

Editorial

I enjoyed GECCO-2009 in Montréal, everything was great as usual. Franz Rothlauf, Pat Cattolico, and the organizing committee really did an amazing job. I was impressed by Stephanie Forest, who swept up both the best paper award in the Genetic Programming track, the Humies gold medal, and the first GECCO Impact Award. This new award focuses on the papers, published at GECCO ten years before, that had the highest impact and number of citations.

The first GECCO impact award went to two papers published in GECCO-99:

- S. Hofmeyer, S. Forrest. Immunity by Design: An Artificial Immune System
- M. Pelikan, D. Goldberg, E. Cantu-Paz. BOA: The Bayesian Optimization Algorithm

I recently read that the paper by Martin Pelikan, David Goldberg and Erik Cantu-Paz was in fact a project from David Goldberg's class Genetic Algorithms in Search, Optimization, and Machine Learning (GE-485) at the University of Illinois at Urbana-Champaign. I wonder what the final grade was. . .

In this issue, with Hans-Paul Schwefel's interview, we continue the series based on the list of 20 questions sent to iconic figures of our field. Then, in the second contributed paper, Natalio Krasnogor gives us an appetizer of the forthcoming "Handbook of Natural Computation" published in the Natural Computing series of Springer, with a short and unorthodox introduction to memetic algorithms. Nicola Beume and Mike Preuss report on the first GECCO-2009 workshop "Learning from failures in evolutionary computation". The lists of recent journal papers and forthcoming events complete the issue.

We have now completed the third volume. We are still two issues behind schedule, but we are catching up so stay tuned.

At the end, as in every editorial I wrote, here is the list of people who made this possible: Hans-Paul Schwefel, Günter Rudolph , Natalio Krasnogor, Nicola Beume, Mike Preuss, Martin V. Butz, Xavier Llorá, Kumara Sastry, Mario Verdicchio, Viola Schiaffonati, and board members Dave Davis and Martin Pelikan. My due thanks go to them.

I hope you like the cover since I designed it.

Pier Luca
September 21st, 2009

SIGEVOlution
Winter 2008, Volume 3, Issue 4

Newsletter of the ACM Special Interest Group on Genetic and Evolutionary Computation.

Contributors to this Issue

Hans-Paul Schwefel
Günter Rudolph
Natalio Krasnogor
Nicola Beume
Mike Preuss

Contents

ISSN: 1931-8499

An Interview with Hans-Paul Schwefel

with an introduction by Günter Rudolph

Hans-Paul Schwefel, Universität Dortmund, hps@udo.edu

The sounds of Sputnik from its orbit around the earth in 1957 were the origin of the young Hans-Paul Schwefel's desire to become a space traveller. In thoughtful preparation he soon began his studies in Aero- and Space Technology at the Technical University Berlin (TUB). But while he was junior assistant at the Hermann-Föttinger Institute for Hydrodynamics at TUB another event changed his plans: In the 1960s difficult multimodal and noisy optimization problems from engineering sciences awaited their solution at the TUB. At that time mathematical models or numerical simulations of these particular problems were not available. As a consequence, the optimization had to be done experimentally with the real object at hardware level. The three students Peter Bienert, Ingo Rechenberg and Hans-Paul Schwefel (HPS) envisioned an automated cybernetic system that alters the object parameters mechanically or electrically, runs the experiment, measures the outcome of the experiment and uses this information in the context of some optimization strategy for the decision how to alter the object parameters of the real object for the experiment in the subsequent iteration. Obvious candidates for the optimization methods in the framework of this early 'hardware-in-the-loop' approach were all kinds of gradient-like descent methods for minimization tasks. But these methods failed. Inspired by lectures on biological evolution they tried a randomized method that may be regarded as the simplest algorithm driven by mutation and selection—a method nowadays known as the (1+1)-Evolution Strategy. This approach was successful and this event may be seen as the trigger that turned the career of HPS towards the emerging scientific field of evolutionary computation (EC).

After diversified episodes of his professional and scientific life he became full professor of Computer Science at the University of Dortmund where he was holder of the chair for Systems Analysis since 1985. From this position he gave birth to the now well-respected conference series on 'Parallel Problem Solving from Nature' (PPSN) and he was the driving force of establishing the 'Collaborative Research Center on Computational Intelligence' at the University of Dortmund. In addition to his numerous pioneering contributions to the field of EC he also served as a valued teacher and mentor for many master's and PhD students, quite a few of them now extending and propagating his work from positions all over the world.

Hans-Paul Schwefel received numerous rewards, most notable probably the IEEE Fellowship in 2005 and the honorary degree of Doctor of Science from The University of Birmingham (UK) in 2007. Although officially retired since February 2006 he is still active in the EC community and regularly seen in his university office. So we may be excited about the things still to come ...

Günter Rudolph, TU Dortmund University

Q Everybody knows the enormous influence you had in our field. Would you summarize the key ideas of evolutionary strategies in 2-3 paragraphs for someone unfamiliar with the field?

The key idea is a biologically plausible internal mechanism for adapting the variation strength since all efficient numerical optimization techniques do have at least some step size control. Eigen's paradox demonstrates that biologists had problems in understanding how mutation rates could adapt for DNA lengths necessary for even simplest reproductive organisms. I found an explanation and used it for the first 'self-adaptive' multimembered evolution strategy in the early 1970s: sufficiently high birth surplus and medium selection pressure and finite life span (or reproduction cycles per inidividual). The result was the well-known (mu,lambda e.g. 10,100) truncation selection EA. Recombination enhances the process, but it works (and in nature first worked) also without it.

Several other principles beyond mutation, recombination, and selection are worth to be studied and used for natural computing, e.g., gene duplication and gene deletion, polyploidy, multicellularity, gender dimorphism, multi-population, and multi-species schemes.

For multiple criteria conditions (the common case in nature) a predatorS-prey model is appropriate. It also helps to avoid idling processors in grid/cloud computing by means of asynchronous birth/mating/death processes.

Q What experiences in school, if any, influenced you to pursue a career in science?

At and after (1959) school, I did not pursue an academic career. It just happened to me. Enchanted by Sputnik 1, I wanted to become an astronaut visiting distant sites in the universe. But because Lufthansa and Air France did not agree in time about pooling their pilot schools (I passed all qualifying examinations), I started studying aero- and space technology. One year later, I could have started a pilot's career, but meanwhile the university studies fascinated me and let me dream of engineering my own spaceship (ask my wife, she will certify that I promised to take her with me into space).

Q Who are the three people whose work inspired you most in your research?

Wernher von Braun (I bought a copy of his dissertation about constructing missiles), Eugen Sänger (I aimed at becoming his assistant; Unfortunately he died shortly before I got my diploma), and Hermann Oberth - I did not miss any of their lectures at the Technical University of Berlin (TUB).

Later on I was fascinated by contemporary topics like cybernetics, bionics, and computers meeting pioneers like Karl Steinbuch (Lernmatrix, artificial neural networks), Heinrich Hertel (my professor for aircraft construction, teaching to look for structures, forms, and movement in nature, e.g., blades of grass and dolphins), and Konrad Zuse (whose Z23 computer was the first one at the TUB; I used it already for my diploma thesis about simulating evolutionary processes on discrete optimization problems in 1965).

Q What are the three books or papers that inspired you most?

1958: Wernher von Braun & Willy Ley: Die Eroberung des Weltraums (The conquest of space)

1959: Teilhard de Chardin: Der Mensch im Kosmos (I had thought it would deal with 'man in space', but it dealt with organic and spiritual evolution. In French the title was 'Le phénomène humain', which I recognized only later. The book was on Rome's index when I bought it in 1963).

1965: Wilhelm Fucks: Formeln zur Macht (That is why I became a futurologist when EAs did not pay - from 1976 to 1985).

Q As a founding father of this field, what is your own view about what evolutionary strategies are? What did you expect them to be?

A means of understanding 'real life' *and* an aid for experimental (later also computational) improvements.

Q What do you like most about evolutionary computation?

That they have become accepted after 30+ years, i.e. during my life, even by many skeptic theoreticians.

Q Conversely, what do you dislike most about evolutionary computation?

That bio-inspiration has gone more and more into the background in search for efficiency instead of insight.

Q What is the biggest open question in the evolutionary computation area?

How to model further important features of organic evolution in search for understanding nature *and* improving algorithms.

Q Where do you see the evolutionary computation community going in the next ten years? Twenty years?

10 years: hyper-meta-'all-weather'-hybridization with classic optimization tools,
20 years: a class-room standard in engineering and management.

Q What are your favorite real-world applications of evolutionary strategies?

Experimental ones (without computers)

Q Your books and papers are sources of inspirations, is there any topic in your books and papers which you hoped people would take more seriously?

People should read carefully, not only cite from hearsay.

Q Which ones are the most misunderstood/misquoted?

There is nothing that has not been misunderstood by at least one 'researcher'.

Q If you could do it again, what would you do differently in your development of the evolutionary computation field?


“Je ne regrette rien!”

Q What new ideas are you (or former students) working on and excited about?


PredatorS-prey models for multicriteria (including constrained), dynamic, noisy situations

Q What new ideas in evolutionary computation are you excited about?


Two-gender models without and with mating selection

Q What books, tangentially related to the field, that you’ve read in the last year did you like the best?


Books of Daniel C. Dennett (Darwin’s Dangerous Idea) and Ernst Mayr (What Evolution Is, What Makes Biology Unique?) about evolution

Q You had many successful PhD students, what is your recipe for PhD success?


Don’t discourage them; ask questions like Socrates; give plenty of rope.

Q Your key advice to a PhD student?


Don’t follow the advice of elders, follow your intuition!

Q What advice would you give to students and beginning researchers who are starting to work in evolutionary computation?


Don’t follow my advice or my example, find your own way!

Q Has thinking about evolution changed your view on things in general?


Yes, indeed.

An Unorthodox Introduction to Memetic Algorithms

Natalio Krasnogor, Interdisciplinary Optimisation Laboratory
The Automated Scheduling, Optimisation and Planning Research Group
School of Computer Science, University of Nottingham, United Kingdom, Natalio.Krasnogor@Nottingham.ac.uk

Memetic Algorithms have become one of the key methodologies behind solvers that are capable of tackling very large, real-world, optimisation problems. They are being actively investigated in research institutions as well as broadly applied in industry. This article provides a very short introduction to Memetic Algorithms and it is a condensed version of a chapter with the same title that will appear in the Handbook of Natural Computing to be published by Springer [35] in which a pragmatic guide to the key design issues underpinning Memetic Algorithms (MA) engineering is overviewed.

Introduction

Memetic Algorithms was the name given by P.A. Moscato[49] to a class of stochastic global search techniques that, broadly speaking, combine within the framework of Evolutionary Algorithms (EAs) the benefits of problem-specific local search heuristics and multi-agent systems. MAs have been successfully applied to a wide range of domains that cover problems in combinatorial optimisation, e.g. [53, 9, 19, 24, 13, 11, 55, 22], continous optimisation, e.g. [23, 51, 48], dynamic optimisation [57, 10, 58], multi-objective optimization [44, 28, 29], etc. It could be argued that, unlike other nature-inspired algorithms such as Ant Colony Optimisation [16], Simulated Annealing[30], Neural Networks[46], Evolutionary Algorithms[26], etc, Memetic Algorithms lack, at their core, a clear natural metaphor. Whether this is a strength or a weakness of the paradigm is a discussion for another time and in the extended version of this article we opted instead to focus first on a pragmatic software engineering presentation of this remarkably malleable search technology and then, near the end of the article, to argue that there is indeed a potentially powerful nature-inspired metaphor that could lead to important new breakthroughs in the field.

Early in the history of EAs' application to real-world problems, it became aparent that a *canonical GA*, namely one using a simple binary representation, n-point crossover, bitwise mutation and fitness proportionate selection, could not possibly compete with tailor made algorithms. This empirical observation resonated well with theoretical (and experimental) studies on the so called "Baldwin effect" and on "Lamarckian evolution" [25, 6, 27, 45, 56, 59, 60] that focused on how learning could affect the process of evolution. Thus, it became apparent that EA's global search dynamics ought to be complemented with local search refinement provided by a suitable hybridisation using problem specific solvers including heuristics, approximate and exact algorithms. Moreover, further theoretical results such as [61] (and similar subsequent work) debunked the idea that effective and efficient "black box" general problem solvers were attainable, and hence gave further impetus to the school of thought that supported, as an essential methodological component, the incorporation of problem (or domain) specific information in EAs.

Domain-specific knowledge was thus added to the EA framework by means of specialised crossover and mutation operators, sophisticated problem specific representations, smart population initialisation, complex fitness functions, (closer to the spirit of MAs) local search heuristics and, when available, approximate and exact methods. More recently, R. Dawkin's concept of "memes" [15] has been gathering pace within the Memetic Algorithms literature as they can be thought of as representing "evolvable" strategies for problem solving thus breaking the mould of a fixed and static domain knowledge captured once during the design of the MAs and left untouched afterwards. Thus Dawkin's memes, and their extensions ([12, 18, 20, 5]), as evolvable search strategies [31, 32, 34, 37, 54, 7] provide a critical link to the possibility of open-ended combinatorial and/or continuous problem solving.

Software development is a process of knowledge acquisition[3] and the development of successful Memetic Algorithms is no different. The popularity behind MAs is more closely related to the relative ease by which a reasonably good solver can be implemented than to any fundamental advantage over other optimisation techniques such as Tabu Search or Simulated Annealing (to name but two). Indeed, any successful nature-inspired search method owes it's popularity not to an intrinsic problem solving feature, which might be absent from a competing method, but rather to the fact that, in spite of obvious design flaws (e.g. large number of parameters, lack of operational theory for their use, etc), they help to structure around them a healthy research and practice milieu. That is, nature-inspired search methods are computational "research programmes", or "research paradigms", in their own right [40]. Thus the question of what are the key components of the Memetic Algorithms research paradigm takes center stage. The literature has a large number of papers in which a variety of methods are classified as Memetic Algorithms. Thus, although the large majority of Memetic Algorithms are instances of Evolutionary Algorithms-Local Search hybrids, numerous MAs are derived from other metaheuristics, e.g., Ant Colony Optimisation (ACO) [41], Particle Swarm [43], Artificial Immune Systems (AIS)[62], etc. What all of these implementations have in common is a *carefully choreographed interplay between (stochastic) global search and local search strategies*. In the remaining parts of this article, we will mention some of the key implementation strategies that, over the course of the years, have (re)appeared in the form of tried and tested algorithmic design solutions to the ubiquitous problem of how to successfully orchestrate global and local search methods in complex search spaces.

A Pattern Language for Memetic Algorithms

In [2] C. Alexander and colleagues introduced, within the context of architecture and urban planning, the concept of "Patterns" and "Pattern Languages":

> In this book, we present one possible pattern language,... The elements of this language are entities called patterns. Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.

A pattern language is then defined as a collection of interrelated patterns, with each and every one of them expressed in a concise, clear and uniform format. The content of a pattern includes at least the following elements [21] (see [35] for a more complete list):

- *The pattern name*, which is a concise handler to refer to both a specific problem and a tried and tested solution.
- *The problem statement* depicting the situation in which the pattern is best applied, that is, the problem that the pattern attempts to provide a solution to, e.g., maintaining pareto front diversity, etc.
- *The solution*, in turn, provides a template on how to approach the solution of the problem to which the pattern is applied. The description in this section is not prescriptive but rather qualitative. As emphasized by Alexander et al. [2], one might reuse a solution under myriads of different shapes, yet the essential core of all those implementations should be easily distinguishable and invariant.
- *The consequences* of applying the pattern. There are no free lunches, hence even when a pattern might be the best (or perhaps only) solution to a given problem, its application might lead to a series of trade-offs.
- *Representative Examples* briefly mentioning cases where the pattern has been used.

Thus a collection of well defined patterns, i.e. a rich pattern language, substantially enhances our ability to communicate solutions to recurring problems without the need to discuss specific implementation details.

The pattern language thus serves the dual purpose of being both a taxonomy of problems and a catalogue of solutions. A reader interested in, for example, finding out about diversity handling strategies for MAs, irrespective of which underlying framework (e.g. Evolutionary, Ant Colony, Artificial Immune System, etc) the MA is being implemented in, can quickly scan the various patterns in the pattern language catalogue, identify the one related to diversity strategies and rapidly gain an idea of the tried and tested approaches, the pattern's motivation and consequences. Furthermore, the reader could then refer to the mentioned literature for concrete, detailed codes and methods. In this article we will only mention some of the key design patterns behind the design of successful Memetic Algorithms. The full description is given, of course, on the accompanying book chapter[35].

Before describing a specific pattern language for Memetic Algorithms, we overview the pseudocodes and flowcharts of some representative MA instances. We resort to reporting these algorithms in exactly the same form found in the originating publication as to emphasise both the *invariants* in their architecture as well as the variety of "decorations" found in the many implementations of Memetic Algorithms. Other examples of "in the wild" MAs can be found in [39].

Memetic improvements have been used in, e.g., Learning Classifier Systems [4] with the top level pseudocode shown in Fig. 1(a). An Estimation of Distribution-like MA, a Compact Memetic Algorithm[47] is shown in 1(b) (for a more recent EDA-MA see [17]) while a Memetic Particle Swarm Optimisation [52] pseudocode is depicted in 1(c). Figures 1(d) and 1(e) show a generic pseudocode of an Ant Colony Optimisation based MA[14] and its explicit use of solution refining strategies (in the form of local search methods) respectively. An example of an Immune System inspired Memetic Algorithm's [62] flowchart is shown in figure 1(f). The key invariant property that is present in the architecture of all these Memetic Algorithms is the combination of a search mechanism operating over (in principle) the entire search space with other search operators focusing on local regions of these search spaces. This key invariant holds true regardless of the nature-inspire paradigm the Memetic Algorithm is derived from or whether it is meant to solve an NP-hard combinatorial problem or a highly complex (e.g. multimodal, nonlinear, multi-dimensional) continuous one.

We argue that the key problem that is addressed by Memetic Algorithms is the balance between global and local search, in other words, the strategy that different nature-inspired paradigms (e.g. ACO, AIS, etc) might need to implement as to benefit from a successful tradeoff between exploration and exploitation. Thus we can define the first top-level pattern as:

Memetic Algorithm Pattern (MAP)[1]

- *Problem statement:* A Memetic Algorithm provides solution patterns for the ubiquitous problem of how to successfully orchestrate a balanced tradeoff between exploring a search space and exploiting available (partial) solutions. It is suitable for the solution of complex problems where standard and efficient (i.e. approximation algorithms, exact algorithms, etc) methods do not exist. The Memetic Algorithm is said to explore the search space through a "global" search technique while exploitation is achieved through "local" search.
- *The solution:* This pattern relies on finding, for a given problem domain, an adequate instantiation of exploration and exploitation. Exploration is performed by "global search" methods usually implemented by means of a population-based nature-inspired method such as Evolutionary Algorithms, Ant Colony Optimisation, Artificial Immune Systems, etc. Exploitation is commonly done through the use of local search methods and domain specific heuristics. The global scale exploration is achieved by, e.g., keeping track of multiple solutions or by virtue of specific "jump" operators that are able to connect distant regions of the search space. The local scale exploitation focuses the search on the vicinity of a given candidate solution.
- *The consequences:* Hybridising a global search method of any kind with local search and/or domain specific heuristics usually results in better end-results but this comes at the expense of increased computational time. The correct tradeoff between exploration and exploitation must be such that were the global searcher given the same total CPU "budget" as the Memetic Algorithm then its solutions would still be worse than those derived from the MA. Needless to say, if the local searcher by itself, or through a naive multi-start shell could achieve the same quality of results as the Memetic Algorithm then the global searcher becomes irrelevant. Another likely consequence is that local search and domain specific heuristics usually result in premature convergence (also called diversity crisis).

```
Procedure GA Cycle
Population = Initialize population
Evaluate(Population)
For it = 1 to NumIterations
Selection(Population)
Offspring=CrossOver(Population)
Mutation(Offspring)
LocalSearch(Offspring)
Population=Replacement(Population,Offspring)
Evaluate(Population)
EndFor
Output : Best individual from Population

Procedure LocalSearch
Input : Population
ForEach individual in Population
        If rand(0, 1) < probLocalSearch
           Apply Rule-wise Local search operators to individual
        EndIf
EndForEach

Output : Population
```

(a)

```
function cMA(psize, gens, rec, ub) : X

  begin
    for i = 1 to l do p[i] := 0.5;
    x := generate(p);
    x := localSearch(x);
    x* := x;
    i := 1;
    repeat
      x: = generate(p);
      x: = localSearch(x);
      if rec then x' = recombine(x*, x);
      else x' := generate(p);
      x' := localSearch(x');
      if f(x') < f(x) then swap(x, x');
      if f(x') > f(x*) then x* = x';
      if ub then update(p, x*, x, psize)
      else update(p, x', x, psize);
      i := i + 1;
    until (i > gens);
    return x*;
  end
```

(b)

Input: N , χ, c_1, c_2, $x_{\min}$, $x_{\max}$ (lower & upper bounds), F (objective function).
Set $t = 0$.
Initialize $x_i^{(t)}, v_i^{(t)} \in [x_{\min}, x_{\max}]$, $p_i^{(t)} \leftarrow x_i^{(t)}$, $i = 1, \ldots, N$.
Evaluate $F(x_i^{(t)})$.
Determine the indices g_i, $i = 1, \ldots, N$.
While (stopping criterion is not satisfied) **Do**
  Update the velocities $v_i^{(t+1)}$, $i = 1, \ldots, N$, according to (1).
  Set $x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}$, $i = 1, \ldots, N$.
  Constrain each particle x_i in $[x_{\min}, x_{\max}]$.
  Evaluate $F(x_i^{(t+1)})$, $i = 1, \ldots, N$.
  If $F(x_i^{(t+1)}) < F(p_i^{(t)})$ **Then** $p_i^{(t+1)} \leftarrow x_i^{(t+1)}$.
  Else $p_i^{(t+1)} \leftarrow p_i^{(t)}$.
  Update the indices g_i.
  When (local search is applied) **Do**
    Choose (according to one of the Schemata 1–3) $p_q^{(t+1)}$, $q \in \{1, \ldots, N\}$.
    Apply local search on $p_q^{(t+1)}$ and obtain a new solution, y.
    If $F(y) < F(p_q^{(t+1)})$ **Then** $p_q^{(t+1)} \leftarrow y$.
  End When
  Set $t = t + 1$.
End While

(c)

```
1  Procedure daemon_actions
2      for each S_k do local_search(S_k) {optional}
3      rank (S_1, ..., S_m) in decreasing order of solution
       quality into (S'_1, ..., S'_m)
4      if (best(S'_1, S_global-best))
5          S_global-best = S'_1
6      end if
7      for μ = 1 to (σ − 1) do
8          for each edge a_rs ∈ S'_μ do
9              τ_rs = τ_rs + (σ − μ) · f(C(S'_μ))
10         end for
11     end for
12     for each edge a_rs ∈ S_global-best do
13         τ_rs = τ_rs + σ · f(C(S_global-best))
14     end for
15 end Procedure
```

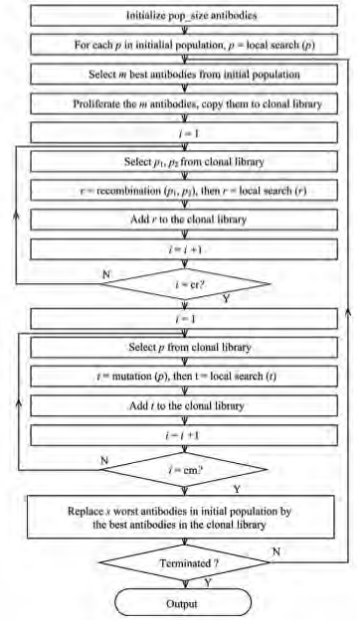
(d)

```
1  Procedure ACO_Metaheuristic
2      parameter_initialization
3      while (termination_criterion_not_satisfied)
4          schedule_activities
5              ants_generation_and_activity()
6              pheromone_evaporation()
7              daemon_actions() {optional}
8          end schedule_activities
9      end while
10 end Procedure


1  Procedure ants_generation_and_activity()
2    repeat in parallel for k=1 to m (number_of_ants)
3        new_ant(k)
4    end repeat in parallel
5  end Procedure


1  Procedure new_ant(ant_id)
2      initialize_ant(ant_id)
3      L = update_ant_memory()
4      while (current_state ≠ target_state)
5          P = compute_transition_probabilities(A,L,Ω)
6          next_state = apply_ant_decision_policy(P,Ω)
7          move_to_next_state(next_state)
8          if (on_line_step-by-step_pheromone_update)
               deposit_pheromone_on_the_visited_edge()
           end if
9          L = update_internal_state()
10     end while
11     if (online_delayed_pheromone_update)
12         for each visited_edge
               deposit_pheromone_on_the_visited_edge()
13     end for
14     end if
15     release_ant_resources(ant_id)
16 end Procedure
```

(e)



(f)

Fig. 1: In (a) the pseudocode (reproduced from [4]) for a Memetic Learning Classifier System, (b) the pseudocode (reproduced from [47]) for an Estimation of Distribution-Like Memetic Algorithm, (c) a Memetic Particle Swarm Optimisation pseudocode (reproduced from [52]), (d) & (e) pseudocode of a representative Ant Colony Optimisation metaheuristic (reproduced from [14]) with a solution refining strategy, through local search, for ACO and (f) an AIS inspired Memetic Algorithm flowchart (reproduced from [62])

- *Examples:* Papers [14, 62, 52, 17, 4] report on the use of Memetic Algorithms under a variety of nature-inspired incarnations.

The MAP can be refined through a variety of Template Patterns (TP)[21], which allow for the definition of the *skeleton* of an algorithm, method or protocol, through deferring problem specific details to subclasses. In this way, through a judicious use of the Template Pattern one can have a very generic and reusable recipe for implementing solutions to a range of, perhaps very different, problems. Figures 1(d) and 1(e), 1(f), 1(c), 1(b) and 1(a) are examples of Template Method Patterns for ACO, AIS, PSO, EDA and LCS based Memetic Algorithms respectively. Each TP captures the invariant properties of Memetic Algorithms when it is implemented from the perspective of a specific nature-inspired algorithms. It also captures the invariant features of MAs regardless of which nature-inspired route is used to implement it, e.g., the entwining of global and local search procedures. In what follows we mention other design patterns that are critical to the implementation of competent Memetic Algorithms thus extending the pattern language for MAs.

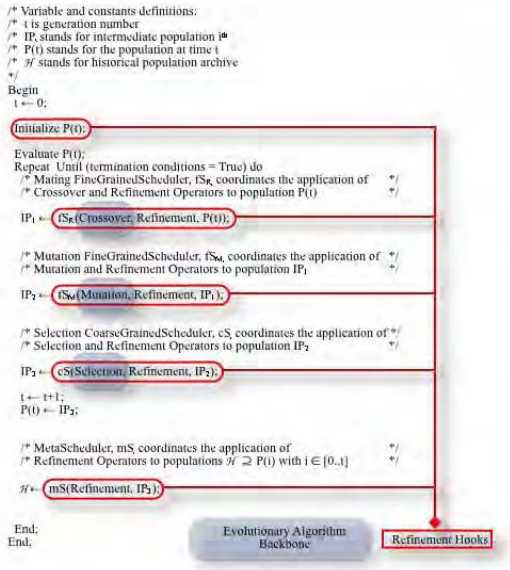


Fig. 2: An Evolutionary Algorithm based Memetic Algorithm Template. The figure highlights the EAs core operators as well as the hotspots where the algorithm can be refined, i.e., where “memetic" operators might come into play

A Template Pattern for an Evolutionary Memetic Algorithm

Evolutionary Memetic Algorithm Template Pattern (EMATP)[2]

- *Problem statement:* the EMATP provides a viable route for solving the problem of how to best coordinate global and local search methods from within an evolutionary algorithms paradigm. Although in principle the simultaneous exploration of the search space by the evolutionary process and the exploitation (by refinement) of candidate solutions should result on an improved algorithm, this is not always the case. The expectation is that a hybridisation of an EA would result in a synergistic net effect that would productively balance local and global search.
- *The solution:* the (almost) standard evolutionary cycle composed of $Initiate \rightarrow Evaluate \rightarrow Mate \rightarrow Mutate \rightarrow Select \rightarrow RepeatUntilDone$ is expanded with domain-specific operators that can refine this cycle. The refinement processes could vary for the different elements of the core EA’s pipeline and could be implemented through exact, approximated or heuristic (e.g. local search) methods. Domain specific operations are often implemented in the form of smart initialisations, local search procedures, etc that refine the input solutions to the Mate and/or Mutate processes as well as (more often) their outputs. Refinements could also be applied to the selection, initialisation and population management processes through e.g. fitness sharing, crowding, population structuring (e.g. cellular/lattice structures, demes, islands, etc) and diversity management strategies.

Figure 2 shows a concrete example of the EMATP. In the figure we have identified the key EA’s components as the “backbone" and the potential places where refinement might take place as refinement hooks. Each hook is represented by a “scheduler" operator that manages the flow of information (e.g. partial/candidate solutions, time-dependent parameters, etc) between each of the core EA’s component and the refinement strategies associated with them. A formal notation for these schedulers can be found in [39]. Needless to say, EMATP can be implemented through a series of (object-oriented) class hierarchies [38]. The EMATP pattern in Fig. 2 can be encapsulated into an abstract class from which subclasses implement specific versions of the pattern. One could thus imagine a family of Evolutionary Memetic Algorithms where one or more features are either removed from the pattern or added to it simply by overriding the behaviour of the scheduler methods.

A Pragmatic Guide to Fitting It All Together

From an analysis of the literature and software available, the following are some of the key patterns defining our language:
At the top of the conceptual hierarchy the Memetic Algorithm Pattern (MAP) appears. Its aim is to provide organisational principles for effective global-local search on hard problems. MAP can be implemented through a number of available (successful) templates some of which are based on Particle Swarm Optimisation, Ant Colony Optimisation, Evolutionary Algorithms, etc. Each of these gives rise to new terms in the language, namely, the EMATP (for the Evolutionary Memetic Algorithm Algorithmic Template) discussed previously has a counter-part for Ant Colony Optimisation, namely, the ACOMATP, one for Simulated Annealing, the SAMATP, etc. Each of these nature-inspired paradigm templates have their own "idiosyncrasies". However, at the time of implementing them as Memetic Algorithms the following common features are captured in new design patterns that help decorate the top level patterns: the Refinement Strategy Pattern (RSP), with focus on heuristic and local search methods and how to employ these within the framework of a global search methodology, the Exact and Approximate Hybridisation Strategy Pattern (EAHSP) that defines ways in which expensive exact (or approximate) algorithms are integrated with a Memetic Algorithms Template Pattern under any nature-inspired realisation. Two other terms in our Pattern Language are the Population Diversity Handling Strategy Pattern (PDHSP) that deals with ways to preserve – in the face of aggressive refinement strategies – global diversity and the Surrogate Objective Function Strategy Pattern (SOFSP) whose role is to define methods for dealing within a Memetic Algorithm with very expensive/noisy/undetermined objective functions. Finally, the Pattern Language presented also include terms for defining the "generation" of Memetic Algorithm one is trying to implement. A Memetic Algorithm of the first generation in which only one refinement strategy is used, is called a Simple Strategy Pattern (SSP) and that is the canonical MAs (e.g. see Figure 2). The Multimeme Strategy Pattern (MSP) and the Self-Generating Strategy Pattern (SGSP), which provide methods for utilising several refinement strategies simultaneously and for synthetising new refinement strategies on the fly respectively, are second and third generation MAs. Thus the resulting Pattern Language has at least the following terms: $\{$ MAP, EMATP, PSOMATP, ACOMATP, SAMATP, ..., AISMATP, RSP, EASHP, CPRSP, PDHSP, SOFSP, SSP, MSP, SGSP $\}$.

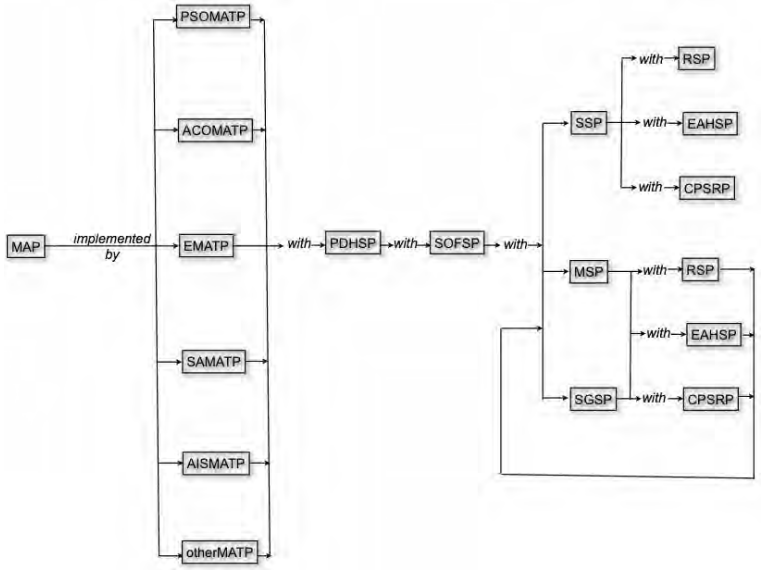


Fig. 3: Integrative view of the design patterns for the Memetic Algorithms Patterns Language. A path through the graph represents a series of design decision that an algorithms engineer would need to take while implementing a MA. Each design pattern provides solutions to specific problems the pattern addresses. By indexing through the patterns' acronyms the reader can have access to examples from the literature where the issues have been solved to satisfaction.

These patterns are functionally related as depicted in Figure 3 that represents the series of design decisions involved in the implementation of a Memetic Algorithms. One could choose to implement a MA by following, e.g., an Ant Colony optimisation template or – more often – an Evolutionary Algorithm template or any of the other template patterns. Each one of these nature-inspired template patterns will have their own algorithmic peculiarities, e.g. crossovers and mutations for EAs, pheromones updating rules for ACOs, etc., but all of them when taken as a Memetic Algorithm will need to address the issues of population diversity, refinement strategies, exact algorithms, surrogate objective functions, single meme versus multimeme[33] versus self-generation[36, 37, 34], etc. In order to instantiate code for any of these design patterns one can simply look at the description of the design pattern provided in this article and refer to any of the literature references given within the pattern description. Thus, Figure 3 provides a reference handbook for MAs engineering.

Conclusions

In this paper we have provided an unorthodox introduction to Memetic Algorithms. We have analysed Memetic Algorithms as they appear in the literature and used in practice, rather than conforming to a top down definition of what is a Memetic Algorithm and then, based on a definition, prescribe how to implement them. The approach taken here allows for a pragmatic view of the field and provides a recipe book for creating Memetic Algorithms by resorting to a shared Pattern Language. This emerging Pattern Language for Memetic Algorithms serves as a catalog of parts (or concerns) that an algorithms engineer might want to resort to for solving specific design issues arising from the need to interlink global search and local search for hard complex problems. Due to space and time constraints, we have not considered design patterns for Multi Objective[8, 42] problems, parallelisation strategies [1, 50], dynamic optimisation, etc., neither how these important new components of the Pattern Language would interact with the patterns described here. It is hoped that the Pattern Language for MAs introduced here and extended in the full article [35] will, in the future, be expanded and refined by researchers and practitioners alike.

Acknowledgement

This research was partially carried out while the author was with the Faculty of Natural Sciences Distinguished Scientist Visitors Program at the Computer Science Department, Ben-Gurion University of the Negev, Israel. In particular he wishes to thank Prof. Moshe Sipper for hosting his visit and his research team for many useful and insightful discussions. The author also acknowledges funding from the EPSRC for projects EP/D061571/1 and EP/C523385/1 and Dr. Pier Luca Lanzi for encouraging him to write this article.

Bibliography

[1] Alba, E., Tomassini, M.: Parallelism and evolutionary algorithms. IEEE Transactions on Evolutionary Computation **6**, 443–462 (2002)

[2] Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., Angel, S.: A Pattern Language - Towns, Buildings, Construction. Oxford University Press (1977)

[3] Armour, P.: The conservation of uncertainty, exploring different units for measuring software. Communications of the ACM **50**, 25–28 (2007)

[4] Bacardit, J., Krasnogor, N.: Performance and efficiency of memetic Pittsburgh learning classifier systems. Evolutionary Computation **17(3)** (2009)

[5] Blackmore, S.: The Meme Machine. Oxford University Press (1999)

[6] Bull, L., Holland, O., Blackmore, S.: On meme–gene coevolution. Artificial Life **6**, 227–235 (2000)

[7] Burke, E., Hyde, M., Kendall, G., Woodward, J.: Automatic heuristic generation with genetic programming: Evolving a jack-of-all-trades or a master of one. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007), pp. 1559–1565. ACM (2007)

[8] Burke, E., Landa-Silva, J.: The design of memetic algorithms for scheduling and timetabling problems. In: W. Hart, N. Krasnogor, J. Smith (eds.) Recent Advances in Memetic Algorithms, pp. 289–312. sv (2004)

[9] Burke, E., Newall, J., Weare, R.: A memetic algorithm for university exam timetabling. In: E. Burke, P. Ross (eds.) The Practice and Theory of Automated Timetabling, *Lecture Notes in Computer Science*, vol. 1153, pp. 241–250. Springer Verlag (1996)

[10] Caponio, A., Cascella, G., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for on-line and off-line control design of pmsm drives. IEEE Transactions On Systems, Man and Cybernetics - Part B **37**, 28–41 (2007)

[11] Carr, R., Hart, W., Krasnogor, N., Burke, E., Hirst, J., Smith, J.: Alignment of protein structures with a memetic evolutionary algorithm. In: W. Langdon, E. Cantu-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. Potter, A. Schultz, J. Miller, E. Burke, N. Jonoska (eds.) GECCO-2002: Proceedings of the Genetic and Evolutionary Computation Conference. Morgan Kaufman (2002)

[12] Cavalli-Sforza, L., Feldman, M.: Cultural Transmission and Evolution: A Quantitative Approach. Princeton University Press (1981)

[13] Cheng, R., Gen, M.: Parallel machine scheduling problems using memetic algorithms. Computers & Industrial Engineering **33**(3–4), 761–764 (1997)

[14] Cordon, O., Herrera, F., Stutzle, T.: A review on the ant colony optimization metaheuristic: Basis, models and new trends. MATHWARE AND SOFT COMPUTING **9** (2002)

[15] Dawkins, R.: The Selfish Gene. Oxford University Press, New York (1976)

[16] Dorigo, M., , Gambardela, L.: Ant colony system: A cooperative learning approach to the travelling salesman problem. IEEE Transactions on Evolutionary Computation **1**(1) (1997)

[17] Duque, T., Goldberg, D., K.Sastry: Improving the efficiency of the extended compact genetic algorithm. In: GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation, pp. 467–468. ACM, New York, NY, USA (2008). DOI http://doi.acm.org/10.1145/1389095.1389181

[18] Durham, W.: Coevolution: Genes, Culture and Human Diversity. Stanford University Press (1991)

[19] Fleurent, C., Ferland, J.: Genetic and hybrid algorithms for graph coloring. Annals of Operations Research **63**, 437–461 (1997)

[20] Gabora, L.: Meme and variations: A computational model of cultural evolution. In: L.Nadel, D. Stein (eds.) 1993 Lectures in Complex Systems, pp. 471–494. Addison Wesley (1993)

[21] Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Desing Patterns, Elements of Reusable Object-Oriented Sofware. Addison-Wesley (1995)

[22] Gutin, G., Karapetyan, D., Krasnogor, N.: Memetic algorithm for the generalized asymmetric traveling salesman problem. In: M. Pavone, G. Nicosia, D. Pelta, N. Krasnogor (eds.) Proceedings of the 2007 Workshop On Nature Inspired Cooperative Strategies for Optimisation. Lecture Notes in Computer Science (LNCS), vol. to appear. Springer (2007)

[23] Hart, W.E.: Adaptive Global Optimization with Local Search. Ph.D. Thesis, University of California, San Diego (1994)

[24] He, L., Mort, N.: Hybrid genetic algorithms for telecomunications network back-up routeing. BT Technology Journal **18**(4) (2000)

[25] Hinton, G., Nowlan, S.: How learning can guide evolution. Complex Systems **1**, 495–502 (1987)

[26] Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press (1976)

[27] Houck, C., Joines, J., Kay, M., Wilson, J.: Empirical investigation of the benefits of partial lamarckianism. Evolutionary Computation 5(1): 31-60. (1997)

[28] Ishibuchi, H., Kaige, S.: Implementation of simple multiobjective memetic algorithms and its application to knapsack problems. Int. J. Hybrid Intell. Syst. **1**(1-2), 22–35 (2004)

[29] Jaszkiewicz, A.: Genetic local search for multi-objective combinatorial optimization. European Journal of Operational Research **137** (2002)

[30] Kirkpatrick, S., Gelatt, C., Vecchi, M.P.: Optimization by simulated annealing. Science **220**, 671–680 (1983)

[31] Krasnogor, N.: Coevolution of genes and memes in memetic algorithms. In: A. Wu (ed.) Proceedings of the 1999 Genetic and Evolutionary Computation Conference Graduate Students Workshop Program (1999). URL http://www.cs.nott.ac.uk/~nxk/PAPERS/memetic.pdf. (poster)

[32] Krasnogor, N.: Studies on the Theory and Design Space of Memetic Algorithms. Ph.D. Thesis, University of the West of England, Bristol, United Kingdom (2002). URL http://www.cs.nott.ac.uk/~nxk/PAPERS/thesis.pdf

[33] Krasnogor, N.: Recent Advances in Memetic Algorithms, *Studies in Fuzziness and Soft Computing*, vol. 166, chap. Towards robust memetic algorithms, pp. 185–207. Springer Berlin Heidelberg New York (2004)

[34] Krasnogor, N.: Self-generating metaheuristics in bioinformatics: the protein structure comparison case. Genetic Programming and Evolvable Machines **5**(2), 181–201 (2004)

[35] Krasnogor, N.: Handbook of Natural Computation, chap. Memetic Algorithms, p. (to appear). Natural Computing. Springer Berlin / Heidelberg (2009). URL http://www.cs.nott.ac.uk/~nxk/PAPERS/hnc-nxk.pdf

[36] Krasnogor, N., Gustafson, S.: Toward truly "memetic" memetic algorithms: discussion and proof of concepts. In: D.Corne, G.Fogel, W.Hart, J.Knowles, N.Krasnogor, R.Roy, J.E.Smith, A.Tiwari (eds.) Advances in Nature-Inspired Computation: The PPSN VII Workshops. PEDAL (Parallel, Emergent and Distributed Architectures Lab). University of Reading. ISBN 0-9543481-0-9 (2002)

[37] Krasnogor, N., Gustafson, S.: A study on the use of "self-generation" in memetic algorithms. Natural Computing **3**(1), 53 – 76 (2004)

[38] Krasnogor, N., Smith, J.: Mafra: A java memetic algorithms framework. In: A. Wu (ed.) Workshop Program,Proceedings of the 2000 Genetic and Evolutionary Computation Conference. Morgan Kaufmann (2000)

[39] Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: Model, taxonomy and design issues. IEEE Transactions on Evolutionary Algorithms **9**(5), 474–488 (2005)

[40] Kuhn, T.: The Structure of Scientific Revolution. University of Chicago Press (1962)

[41] Lee, Z., Lee, C.: A hybrid search algorithm with heuristics for resource allocation problem. Information Sciences **173**, 155–167 (2005)

[42] Li, H., Landa-Silva, D.: Evolutionary multi-objective simulated annealing with adaptive and competitive search direction. In: Proceedings of the 2008 IEEE Congress on Evolutionary Computation (CEC 2008), pp. 3310–3317. IEEE Press (2008)

[43] Liu, B.F., Chen, H.M., Chen, J.H., Hwang, S.F., Ho, S.Y.: Meswarm: memetic particle swarm optimization. In: GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation, pp. 267–268. ACM, New York, NY, USA (2005). DOI http://doi.acm.org/10.1145/1068009.1068049

[44] Liu, D., Tan, K.C., Goh, C.K., Ho, W.K.: A multiobjective memetic algorithm based on particle swarm optimization. Systems, Man, and Cybernetics, Part B, IEEE Transactions on **37**(1), 42–50 (2007). DOI 10.1109/TSMCB.2006.883270

[45] Mayley, G.: Landscapes, learning costs and genetic assimilation. Evolutionary Computation **4**(3), 213–234 (1996)

[46] McCulloch, W., Pitts, W.: A logical calculus of the ideas immanent in nervous system. Bulletin of Mathematical Biophysics **5**, 115–133 (1943)

[47] Merz, P.: The compact memetic algorithm. In: Proceedings of the IV International Workshop on Memetic Algorithms (WOMA IV). GECCO 2003. http://w210.ub.uni-tuebingen.de/portal/woma4/ (2003)

[48] Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a lamarkian genetic algorithm and an empirical binding free energy function. J Comp Chem **14**, 1639–1662 (1998)

[49] Moscato, P.: On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Tech. Rep. Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA (1989)

[50] Nebro, A., Alba, E., Luna, F.: Multi-objective optimization using grid computing. Soft Computing **11**, 531–540 (2007)

[51] Niesse, J., Mayne, H.: Global geometry optimization of atomic clusters using a modified genetic algorithm in space-fixed coordinates. Journal of Chemical Physics **105**(11), 4700–4706 (Sep. 15, 1996)

[52] Petalas, Y., Parsopoulos, K., Vrahatis, M.: Memetic particle swarm optimisation. Annals of Operations Research **156**, 99–127 (2007)

[53] Reeves, C.: Hybrid genetic algorithms for bin-packing and related problems. Annals of Operations Research **63**, 371–396 (1996)

[54] Smith, J.: Co-evolving memetic algorithms: A learning approach to robust scalable optimisation. In: Proceedings of the 2003 Congress on Evolutionary Computation, pp. 498–505 (2003)

[55] Tang, M., Yao, X.: A memetic algorithm for VLSI floorplanning. Systems, Man, and Cybernetics, Part B, IEEE Transactions on **37**(1), 62–69 (2007). DOI 10.1109/TSMCB.2006.883268

[56] Turney, P.: How to shift bias: lessons from the Baldwin effect. Evolutionary Computation **4**(3), 271–295 (1996)

[57] Vavak, F., Fogarty, T.: Comparison of steady state and generational genetic algorithms for use in nonstationary environments. In: Proceedings of the 1996 IEEE Conference on Evolutionary Computation, pp. 192–195 (1996)

[58] Wang, H., Wang, D., Yang, S.: A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems. Soft Computing **to appear** (2008)

[59] Whitley, L., Gordon, S., Mathias, K.: Lamarkian evolution, the Baldwin effect, and function optimisation. In: Y. Davidor, H.P. Schwefel, R. Männer (eds.) PPSN, *Lecture Notes in Computer Science*, vol. 866, pp. 6–15. Springer (1994)

[60] Whitley, L., Gruau, F.: Adding learning to the cellular development of neural networks: evolution and the Baldwin effect. Evolutionary Computation **1**, 213–233 (1993)

[61] Wolpert, D., Macready, W.: No Free Lunch theorems for optimisation. IEEE Transactions on Evolutionary Computation **1**(1), 67–82 (1997)

[62] Yanga, J., Suna, L., Leeb, H., Qiand, Y., Liang, Y.: Clonal selection based memetic algorithm for job shop scheduling problems. Journal of Bionic Engineering **5**, 111–119 (2008)

About the author

Natalio Krasnogor is an Associate Professor within the Automated Scheduling, Optimisation and Planing Research Group (ASAP) within the School of Computer Science at the University of Nottingham. He established and co-chairs the series of international workshops on Memetic Algorithms (WOMA), Nature-Inspired Cooperative Strategies for Optimisation (NICSO), Embodied Evolution (EmboEvo) and the European Conference on Synthetic Biology (ECSB). Dr Krasnogor has published more than 70 refereed papers. He was a guest editor for the Evolutionary Computation Journal, Journal of Fuzzy Sets and Systems, IEEE Transactions on Systems Man and Cybernetics and BioSystems Journal. He is editor of the book Systems Self-Assembly: Multidisciplinary Snapshots to appear in Elseviers Studies in Multidisciplinarity series and edited a book on Memetic Algorithms for Springer. He is associated editor for the Evolutionary Computation journal, Founding Chief-Editor (technical) of the new Springers Journal Memetic Computing and editor for the journal of Simulation and Modelling in Engineering and Journal of Artificial Evolution and Applications. He co-chairs the IEEE Computational Intelligent Society's Emergent Technology Technical Committee on Molecular Computing. He is investigator and co-investigator in grants totalling ĩ2M from EPSRC, BBSRC and the EU. Krasnogor is a member of the EPSRC peer review college and also a member of Nottinghams Centre for Plant Integrative Biology (www.cpib.eu), one of the 6 EPSRC/BBSRC flagship centres for Systems Biology. He currently supervises 4 Postdoctoral fellows and 9 Ph.D. students.

Homepage: http://www.cs.nott.ac.uk/~nxk
Email: Natalio.Krasnogor@Nottingham.ac.uk

Events Reports

Learning from Failures in Evolutionary Computation @ GECCO-2009

Mike Preuss and Nicola Beume, TU Dortmund, Germany
Workshop webpage: [WWW]
Conference webpage: http://www.sigevo.org/gecco-2009

Why should one do a workshop on failures when the primary goal of each scientific investigation is success? Looking at former and recent publications in evolutionary computation, the general impression is indeed that only success stories are told. As everybody who is experienced in experimental or theoretical research knows that failures happen, and are sometimes even necessary steps to success, the question shall be raised if reporting failures does have a positive impact on further research on a similar topic. Most researchers who gave feedback to our call agreed that it does. However, there is currently no methodology or silent consent on which failures to report and how. Furthermore, it is even unclear how much different frequently occurring failures are. As this was to our knowledge the first workshop on this topic in evolutionary computation, the most important task was thus to collect a number of cases and get an overview of the predominant types of failures.

The workshop had a total of 6 submissions, of which 4 were accepted. In addition to the paper presentations, two invited speakers gave very interesting insider information on some of their own failures: Hans-Paul Schwefel and Pier Luca Lanzi.

Hans-Paul Schwefel demonstrated three failures he experienced in his early career, and he came up with simple yet intriguing rules how to avoid them.

- Restricting discrete mutations to positions in the neighborhood of the parents only, sooner or later leads to premature stagnation. *Using wider spread distributions — e.g. geometrical for discrete and gaussian ones for continuous parameter optimization solves the problem.*
- Experimenting with too few degrees of freedom - e.g. only the two opening angles for the convergent and divergent parts of the Laval type hot water flashing nozzle (Figure 1) prevents finding the true optimum. 'Expect the unexpected' should be the advice permitting to detect surprising results, maybe even asking for new theories. Expected for the nozzle was that traditional theory holds and the throat area alone determines the mass flow - which turned out not to be the case due to nonequilibrium flow. The first experiments therefore led to maximizing the mass flow instead of (only) the efficiency (thrust / mass flow). For the second series of experiments the mass flow had to be measured in addition to the thrust. *Changing all variables at a time gives often required degrees of freedom, and performing the experiment to see if an idea works even if theory tells otherwise can be successful and even have an impact on theory later on.*
- Using evolution strategies for numerical optimization needed self-adaptive mutation strengths in order to compete with classical methods. Achieving self-adaptation after many trials and errors was found to require births surplus, limited life span, and more than one surviving offspring. Too strong selection pressure evokes individuals that like to take the easy way out in lower dimensional subspaces where they can be faster for a while until they reach the relative optimum and need completely new orientation. *Self-adaptation is a collective achievement.*

The presented papers touched different types of failures, from the unexpected failure of a general idea in *On the Limitations of Adaptive Resampling using the Student's t-test in Evolution Strategies* by Johannes W. Kruisselbrink, Michael T.M. Emmerich, and Thomas Bäck to the struggling with different hardnesses to finally get an idea working in *Reinforcement Learning for Games: Failures and Successes* by Wolfgang Konen and Thomas Bartz-Beielstein. A work in another field, but also given as chronology of subsequent failures was *A Series of Failed and Partially Successful Fitness Functions for Evolving Spiking Neural Networks* by J. David Schaffer, Heike Sichtig, and Craig Laramee.

Fig. 1: Hans-Paul Schwefel setting up the two-phase nozzle experimental optimization at TU Berlin (some time ago)

This type of extended research path description was also advocated by the workshop chairs in the discussion and the closing talk: It could be very useful for other researchers—or oneself years later—to document at least in short what unexpectedly failed on the path to success. The fourth paper by Jörn Mehnen, *Lessons Learned in Evolutionary Computation: 11 Steps to Success* put more emphasis on general experiences with failures in evolutionary computation and how to avoid them on a methodological level and is probably one of the first of this kind.

In the last years, competitions in different areas of evolutionary computation have attained more and more attention and have become an important part of GECCO conferences. Global optimization, music&arts and especially games are the most popular topics. It is interesting to see that so many people are highly motivated to have their methods tested by an independent jury while knowing that only few of them will be successful and most will fail.

Pier Luca Lanzi gave an invited talk with the title *Learning from Failures – Experiences from Game Competitions* as representative of a group of authors consisting of himself, Simon Lucas, and Julian Togelius. Besides losing the competition, there are surprisingly many ways to fail. From the organizers viewpoint, the first possible failure is the interface. If it turns out to consume too much time to get things running, many potential participators will fail on this level already. However, if the interface is simple and there are no constraints concerning computational resources, the entries may get very similar (feature diffusion) over time. It may also turn out that the game itself does not pose the most challenging task but that preprocessing the data is more important, as happened in the Ms Pac-Man competition where screen capture processing became the crucial factor for beating competitors. As an interesting example for one of the first bots learning online from their failures, the COBOSTAR car racing controller of Michael Butz and others was presented. It keeps an internal track model and avoids driving fast at locations where it crashed once.

The workshop chairs consider competitions very important with regard to failures as they deliberately make problems and rankings of submissions public. They also provide some continuity to and motivation for competitive or cooperative development of algorithms and software over the years and at the same time speed up turnaround cycles at least for competitions held more than once a year.

Next to the presentations, a short debate was held over the importance of failures, and how to learn from them. Among others, the question was raised if failures could be used to demonstrate why another (possibly more randomized) approach is needed instead of a standard technique. It was also discussed if at all and how partial or full failures shall be published.

Having many good presentations and lively discussions following these, the workshop was a great success. However, the failures issue is not 'resolved' yet, there are many ways to continue walking this path. Future work shall include collecting more complete failures and failures to success stories in order to set up a taxonomy of failures.

For next year, there are plans to set up a second issue of this GECCO workshop again, possibly in collaboration with Thomas Stützle and Mauro Birattari. The interested reader is invited to provide ideas or views in the workshop context to the authors. Further information is available at:

http://ls11-www.cs.uni-dortmund.de/people/beume/failures.jsp

Failures in other scientific areas are published in different ways by more and more communities. A list of links can be found here: http://jinr.site.uottawa.ca/links.htm (many thanks to Joshua Knowles for the reference).

Besides the invited speakers, the submitting authors, and the interested audience, we are indebted to the program committee, namely Carlos Fonseca, Thomas Jansen, and Thomas Stützle.

New Issues of Journals

Evolutionary Computation 17(3) (www)

- **Evolutionary Undersampling for Classification with Imbalanced Datasets: Proposals and Taxonomy**, Salvador García, Francisco Herrera (pdf)
- **Performance and Efficiency of Memetic Pittsburgh Learning Classifier Systems**, Jaume Bacardit, Natalio Krasnogor (pdf)
- **Estimating the Ratios of the Stationary Distribution Values for Markov Chains Modeling Evolutionary Algorithms**, Boris Mitavskiy, Chris Cannings (pdf)
- **Localization for Solving Noisy Multi-Objective Optimization Problems**, Lam T. Bui, Hussein A. Abbass, Daryl Essam (pdf)
- **A Preference-Based Evolutionary Algorithm for Multi-Objective Optimization**, Lothar Thiele, Kaisa Miettinen, Pekka J. Korhonen, Julian Molina (pdf)
- **Locating and Characterizing the Stationary Points of the Extended Rosenbrock Function**, Schalk Kok, Carl Sandrock (pdf)

Genetic Programming and Evolvable Machines 10(3) (www)

- **A three-step decomposition method for the evolutionary design of sequential logic circuits**, Houjun Liang, Wenjian Luo and Xufa Wang, pp 231-262 (pdf)
- **Evolutionary design of Evolutionary Algorithms**, Laura Diosan and Mihai Oltean, pp 263-306 (pdf)
- **Semantic analysis of program initialisation in genetic programming**, Lawrence Beadle and Colin G. Johnson, pp 307-337 (pdf)

Calls and Calendar

July 2010

GECCO 2010 - Genetic and Evolutionary Computation Conference
July 7-10, 2010, Portland, Oregon, USA
Homepage: http://www.sigevo.org/gecco-2010
Deadline: January 13, 2010
Author notification: March 10, 2010
Camera-ready: April 5, 2010

The Genetic and Evolutionary Computation Conference (GECCO-2010) will present the latest high-quality results in the growing field of genetic and evolutionary computation.

Topics include: genetic algorithms, genetic programming, evolution strategies, evolutionary programming, real-world applications, learning classifier systems and other genetics-based machine learning, evolvable hardware, artificial life, adaptive behavior, ant colony optimization, swarm intelligence, biological applications, evolutionary robotics, coevolution, artificial immune systems, and more.

Organizers

General Chair:	Martin Pelikan
Editor-in-Chief:	Jürgen Branke
Local Chair:	Kumara Sastry
Publicity Chair:	Pier Luca Lanzi
Tutorials Chair:	Una-May O'Reilly
Workshops Chair:	Jaume Bacardit
Competitions Chairs:	Christian Gagné
Late Breaking Papers Chair:	Daniel Tauritz
Graduate Student Workshop	Riccardo Poli
Business Committee:	Erik Goodman
	Una-May O'Reilly
EC in Practice Chairs:	Jörn Mehnen
	Thomas Bartz-Beielstein,
	David Davis

Important Dates

Paper Submission Deadline	January 13, 2010
Decision Notification	March 10, 2010
Camera-ready Submission	April 5, 2010

Venue

The Portland Marriott Downtown Waterfront Hotel, located in downtown Portland, is near the Portland Riverplace Marina, restaurants, shopping & performing arts venues. Hotel room conference rate $179 includes complimentary in-room high-speed Internet access.

More Information

Visit www.sigevo.org/gecco-2010 for information about electronic submission procedures, formatting details, student travel grants, the latest list of tutorials and workshop, late-breaking papers, and more.

For technical matters, contact Conference Chair Martin Pelikan at pelikan@cs.umsl.edu.

For conference administration matters contact Primary Support Staff at gecco-admin@tigerscience.com.

GECCO is sponsored by the Association for Computing Machinery Special Interest Group for Genetic and Evolutionary Computation.

2010 IEEE World Congress on Computational Intelligence

July 18-23, 2010, Barcelona, Spain

Homepage: WWW

Deadline: January 31, 2010

The 2010 IEEE World Congress on Computational Intelligence (IEEE WCCI 2010) is the largest technical event in the field of computational intelligence. It will host three conferences: the 2010 International Joint Conference on Neural Networks (IJCNN 2010), the 2010 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2010), and the 2010 IEEE Congress on Evolutionary Computation (IEEE CEC 2010). IEEE WCCI 2010 will be held in Barcelona, a Mediterranean city located in a privileged position on the northeastern coast of Spain. Barcelona combines history, art, architecture, and charm within a pleasant, and efficient urban environment where meet old friends, and make new ones. The congress will provide a stimulating forum for scientists, engineers, educators, and students from all over the world to discuss and present their research findings on computational intelligence.

Important Due Dates

- Submission deadline: January 31, 2010
- Competition proposals: November 15, 2009
- Special sessions proposals: December 13, 2009
- Notification of special session acceptance: December 22, 2009
- Paper submission: January 31, 2010
- Tutorial and workshop proposal: February 14, 2010
- Notification of tutorial and workshop acceptance: February 22, 2010
- Notification of paper acceptance: March 15, 2010
- Final paper submission: May 2, 2010
- Early registration: May 23, 2010
- Tutorial and Workshops: July 18, 2010
- IEEE WCCI 2010 Conference: July 19, 2010

For more information visit http://www.wcci2010.org/call-for-papers

September 2010

Seventh International Conference on Swarm Intelligence

September 8-10, 2010. Brussels, Belgium

Homepage: http://iridia.ulb.ac.be/ants2010

Deadline: February 28, 2010

Swarm intelligence is a relatively new discipline that deals with the study of self-organizing processes both in nature and in artificial systems. Researchers in ethology and animal behavior have proposed many models to explain interesting aspects of social insect behavior such as self-organization and shape-formation. Recently, algorithms and methods inspired by these models have been proposed to solve difficult problems in many domains.

An example of a particularly successful research direction in swarm intelligence is ant colony optimization, the main focus of which is on discrete optimization problems. Ant colony optimization has been applied successfully to a large number of difficult discrete optimization problems including the traveling salesman problem, the quadratic assignment problem, scheduling, vehicle routing, etc., as well as to routing in telecommunication networks. Another interesting approach is that of particle swarm optimization, that focuses on continuous optimization problems. Here too, a number of successful applications can be found in the recent literature. Swarm robotics is another relevant field. Here, the focus is on applying swarm intelligence techniques to the control of large groups of cooperating autonomous robots.

ANTS 2010 will give researchers in swarm intelligence the opportunity to meet, to present their latest research, and to discuss current developments and applications.

The three-day conference will be held in Brussels, Belgium, on September 8-10, 2010. Tutorial sessions will be held in the mornings before the conference program.

Relevant Research Areas

ANTS 2010 solicits contributions dealing with any aspect of swarm intelligence. Typical, but not exclusive, topics of interest are:

- Behavioral models of social insects or other animal societies that can stimulate new algorithmic approaches.
- Empirical and theoretical research in swarm intelligence.
- Application of swarm intelligence methods, such as ant colony optimization or particle swarm optimization, to real-world problems.
- Theoretical and experimental research in swarm robotics systems.

Publication Details As for previous editions of the ANTS conference, proceedings will be published by Springer in the LNCS series (to be confirmed). The journal Swarm Intelligence will publish a special issue dedicated to ANTS 2010 that will contain extended versions of the best research works presented at the conference.

Best Paper Award

A best paper award will be presented at the conference.

Further Information

Up-to-date information will be published on the web site http://iridia.ulb.ac.be/ants2010/. For information about local arrangements, registration forms, etc., please refer to the above-mentioned web site or contact the local organizers at the address below.

Conference Address

ANTS 2010
IRIDIA CP 194/6 — Tel +32-2-6502729
Université Libre de Bruxelles — Fax +32-2-6502715
Av. F. D. Roosevelt 50 — http://iridia.ulb.ac.be/ants2010
1050 Bruxelles, Belgium — email: ants@iridia.ulb.ac.be

Important Dates

Submission deadline	March 28, 2010
Notification of acceptance	April 30, 2010
Camera ready copy	May 14, 2010
Conference	September 8–10, 2010

April 2010

Evostar 2010 - EuroGP, EvoCOP, EvoBIO and EvoWorkshops
April 7-9, 2010, Istanbul Technical University, Istanbul, Turkey
Homepage: www.evostar.org
Deadline: November 4, 2009

The EuroGP, EvoCOP, EvoBIO and EvoApplications conferences compose EVO*: Europe's premier co-located events in the field of Evolutionary Computing.

Featuring the latest in theoretical and applied research, EVO* topics include recent genetic programming challenges, evolutionary and other meta-heuristic approaches for combinatorial optimisation, evolutionary algorithms, machine learning and data mining techniques in the biosciences, in numerical optimisation, in music and art domains, in image analysis and signal processing, in hardware optimisation and in a wide range of applications to scientific, industrial, financial and other real-world problems.

EVO* Poster

You can download the EVO* poster advertisement in PDF format here (Image: Pelegrina Galathea, by Stayko Chalakov (2009))

EVO* Call for Papers

You can download the EVO* CfP in PDF format here.

EuroGP

13th European Conference on Genetic Programming

EvoCOP

10th European Conference on Evolutionary Computation in Combinatorial Optimisation

EvoBIO

8th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics

EvoApplications 2010

European Conference on the Applications of Evolutionary Computation

- EvoCOMNET: 7th European Event on the Application of Nature-inspired Techniques for Telecommunication Networks and other Parallel and Distributed Systems
- EvoCOMPLEX (new): Evolutionary Algorithms and Complex Systems
- EvoENVIRONMENT: Nature Inspired Methods for Environmental Issues
- EvoFIN: 4th European Event on Evolutionary and Natural Computation in Finance and Economics
- EvoGAMES: 2nd European event on Bio-inspired Algorithms in Games
- EvoIASP: EC in Image Analysis and Signal Processing
- EvoINTELLIGENCE: Nature Inspired Methods for Intelligent Systems
- EvoMUSART: 8th European event on Evolutionary and Biologically Inspired Music, Sound, Art and Design
- EvoNUM: 3rd European event on Bio-inspired algorithms for continuous parameter optimisation
- EvoSTOC: 7th European event on Evolutionary Algorithms in Stochastic and Dynamic Environments
- EvoTRANSLOG: 4th European Event on Evolutionary Computation in Transportation and Logistics

EvoPHD

5th European Graduate Student Workshop on Evolutionary Computation

Evo* Coordinator: Jennifer Willies, Napier University, United Kingdom
j.willies@napier.ac.uk

Local Chair: Şima Uyar, Istanbul Technical University, Turkey
etaner@itu.edu.tr

Publicity Chair: Stephen Dignum, University of Essex, United Kingdom
sandig@essex.ac.uk

About the Newsletter

SIGEVOlution is the newsletter of SIGEVO, the ACM Special Interest Group on Genetic and Evolutionary Computation.

To join SIGEVO, please follow this link [WWW]

Contributing to SIGEVOlution

We solicit contributions in the following categories:

Art: Are you working with Evolutionary Art? We are always looking for nice evolutionary art for the cover page of the newsletter.

Short surveys and position papers: We invite short surveys and position papers in EC and EC related areas. We are also interested in applications of EC technologies that have solved interesting and important problems.

Software: Are you are a developer of an EC software and you wish to tell us about it? Then, send us a short summary or a short tutorial of your software.

Lost Gems: Did you read an interesting EC paper that, in your opinion, did not receive enough attention or should be rediscovered? Then send us a page about it.

Dissertations: We invite short summaries, around a page, of theses in EC-related areas that have been recently discussed and are available online.

Meetings Reports: Did you participate to an interesting EC-related event? Would you be willing to tell us about it? Then, send us a short summary, around half a page, about the event.

Forthcoming Events: If you have an EC event you wish to announce, this is the place.

News and Announcements: Is there anything you wish to announce? This is the place.

Letters: If you want to ask or to say something to SIGEVO members, please write us a letter!

Suggestions: If you have a suggestion about how to improve the newsletter, please send us an email.

Contributions will be reviewed by members of the newsletter board.

We accept contributions in LaTeX, MS Word, and plain text.

Enquiries about submissions and contributions can be emailed to editor@sigevolution.org.

All the issues of SIGEVOlution are also available online at www.sigevolution.org.

Notice to Contributing Authors to SIG Newsletters

By submitting your article for distribution in the Special Interest Group publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance by the editor
- to digitize and post your article in the electronic version of this publication
- to include the article in the ACM Digital Library
- to allow users to copy and distribute the article for noncommercial, educational or research purposes

However, as a contributing author, you retain copyright to your article and ACM will make every effort to refer requests for commercial use directly to you.